AI 응용시스템의 이해와 구축

5강. Advanced Labeling Techniques & Feature Transformation Examples

출석.

Final Project Discussion

잘 준비 되어 가시나요?

프로젝트	팀원
의료진단	**권용순,** 조우성, 박대혁 (Option A)
제품 리뷰 (긍정/부정)판독	**박주형, 정운국, 김용욱**
초해상화	**정태훈, 안혜영, 조대선**
상품 추천	**노용문, 박준호, 한진웅, 이용정 (4 people team)**
제조공정 불량 판정	**임향빈, 오동규, 신용주, 정근시 (4 people team)**

Advanced Labeling Techniques

Data Labeling Methods (3강)

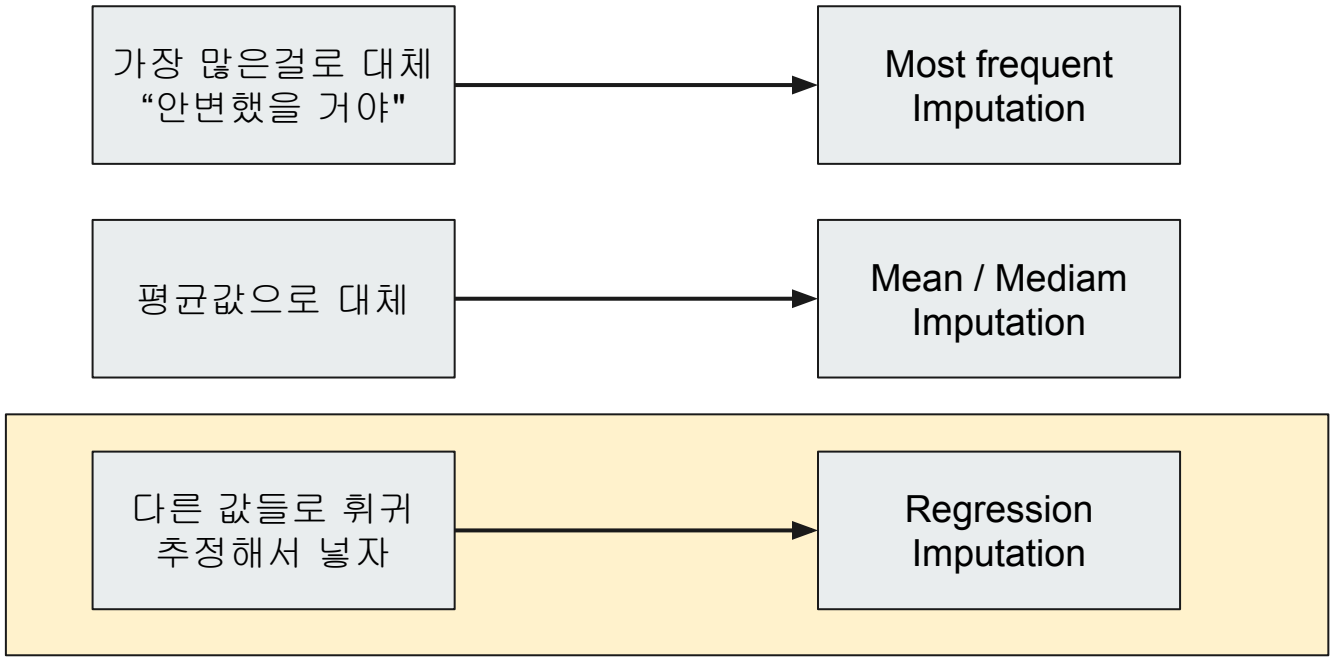
어떤 타입의 라벨링 방법이 있나?

- **피드백 라벨링 (Feedback Labeling)**
  - 시스템에 생성되는 유저피드백으로 라벨링
  - 종종 Derived Label을 이용하여 진행.
- **휴먼 라벨링 (Human Labeling)**

- Semi-supervised labeling
- Active learning
- Weak Supervision

Single Imputation (4강)

누락값(Missing value)들이 있으면 대체값을 어떻게 구할까?

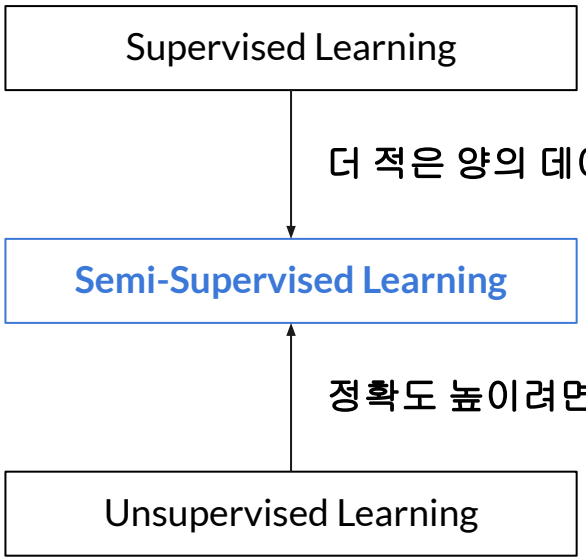


데이터에 따른 머신 러닝 종류

- **Supervised Learning**: 레이블된 데이터로 학습
- **Unsupervised Learning**: 레이블 되지 않은 데이터로 학습
  - k-means, clustering, …
- **Semi-supervised Learning**: 레이블된 일부 데이터와 레이블 되지 않은 데이터 함께 학습에 사용
  - **Active Learning**: 레이블된 **적은 데이터**로 학습 시작, 학습에 중요할것 같은 데이타에 레이블링 제안

데이터에 따른 머신 러닝 종류

그래서 레이블링된 데이터를 Crowd sourcing 하는 제품들이 등장..!
Bobidi (Playstore)



레이블된 데이터 필수
(하지만 **전문성**이 있는 누군가가 레이블링 해야함: **비싸고**, **오래** 걸리고)

더 적은 양의 데이터로 어떻게 안될까?

정확도 높이려면 레이블된 데이터가 필요하겠구나

사람의 수작업 거의 필요없이 바로 모델링 할 수 있지만, 정확도 떨어짐

Semi-supervised learning (준지도학습)

적은 양의 레이블된 데이터와 많은 양의 레이블되지 않은 데이터를 함께 학습에 이용

- 머신러닝 응용분야가 다양해지면서 레이블링도 전문성 요구
- 단순히 레이블링하는 사람 수를 늘리는것만으로는 부족
  - '데이터 레이블링' 작업에는 많은 자원과 비용 필요

- **목표:** 레이블 되지 않은 데이터가 레이블된 데이터의 도움을 받아 학습 정확도 향상

Semi-supervised learning

하지만 SSL이 모든 데이터에서 모델의 성능을 올려주지는 않는다.

레이블 되지 않은 데이터가 의미있게 사용되기 위해서는 몇 가지 가정 필요:

Continuity Assumption	서로 **"가까운"** 인풋 데이터가 동일한 아웃풋을 갖고 있을 확률이 높다	x1 ~ x2 ➔ y1 ~ y2
Cluster Assumption	데이터는 클러스터들을 형성하는 경향이 있고, 동 클러스터 내의 데이터들은 동일한 속성을 갖을 확률이 높다	Decision boundary는 저밀도 지역에 존재한다.
Manifold Assumption	고차원의 인풋 데이터를 저차원적으로 표현할 수 있다	n차원의 인풋데이터가, 사실은 m <<< n차원의 manifold에 존재 ➔ manifold의 구조를 학습하여 예측

Semi-supervised learning

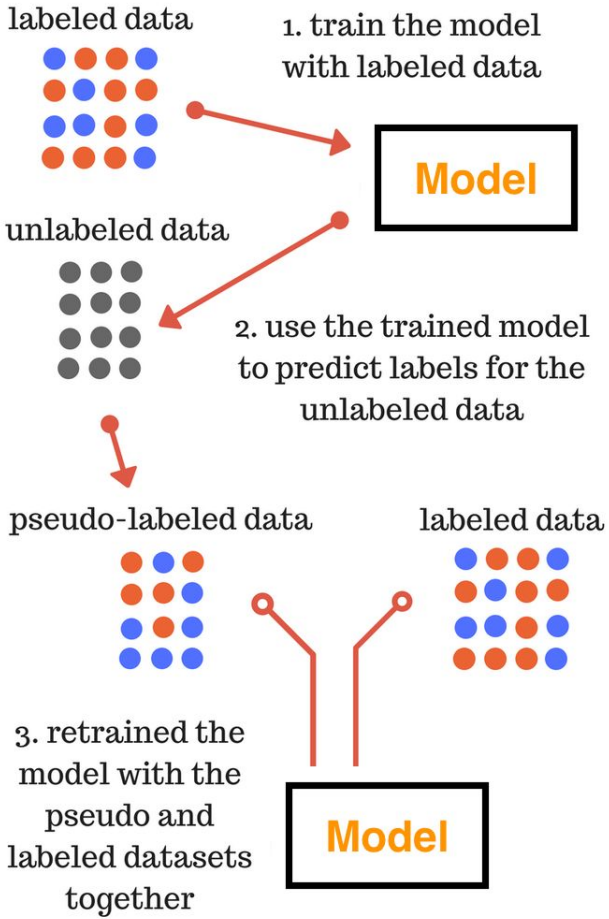
예: web document classifier (뉴스, 쇼핑, 블로그, 여행, ...)

- 제대로 레이블된 **많은 양**의 web document 구하는것은 거의 불가능
- 누군가 모든 document를 읽으며 레이블링 하는것은 매우 비효율적

Methods for SSL: Proxy-label method

Label된 데이터로 학습된 모델을 사용해, unlabeled 데이터를 라벨링하는 기법.

- classification , regression 모두 사용 가능
- Unlabel된 데이터가 Label된 데이터의 distribution을 따르는



Methods for SSL: Entropy Minimization

가정: 모델의 decision boundary는 데이터의 저밀도 지역에 존재

➔ unlabeled 데이터의 entropy를 최소화하려는 방법.

예) Classification Model 두개의 예측 값:

Model A: [**0.8**, 0.1, 0.1] Model B: [**0.6**, 0.2, 0.2]

이때 Model A가 Model B보다 entropy가 낮으므로 채택.

Semi-supervised Learning by Entropy Minimization, NIPS 2004, Grandvalet and Bengio

Active learning

Semi-supervised learning의 특별한 예

- 레이블된 적은 데이터로 학습 시작
- 모델이 레이블 되지 않은 데이터 중 학습에 중요할것 같은 데이터를 **Oracle** (레이블링 하는 사람)에게 레이블링을 요구
- 반복하며 모델 성능 향상

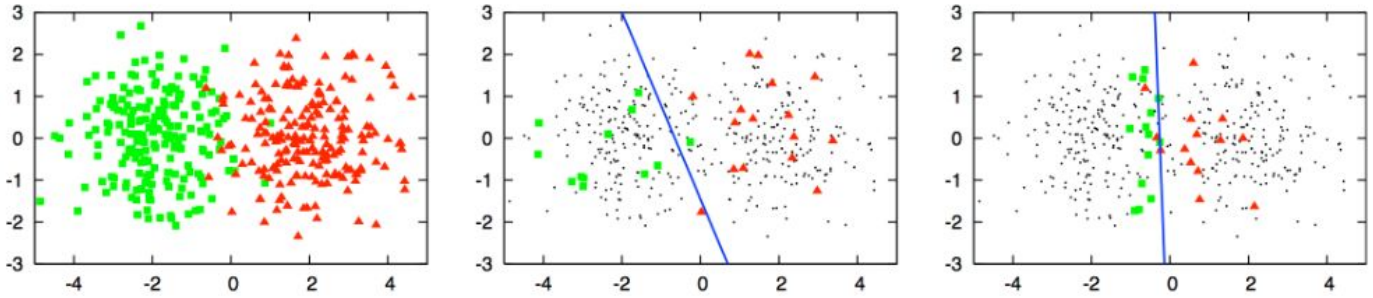
Active learning

예1: 레이블 되지 않은 데이터 (고양이와 호랑이 사진들)

- 몇 개의 데이터만 레이블링, 학습 진행
- 학습 결과, decision boundary에 있는 데이터 추출 (호랑이같은 고양이)
- **Oracle** (예: 레이블링 하는 사람)에게 레이블링 해달라고 함
- 반복하며 성능 향상

신뢰도가 낮은 어려운 데이터를 찾아서 레이블링 하면 모델 성능이 좋아진다는 가정

Active learning



예2)

- 왼쪽: 모든 데이터의 라벨
- 가운데: 소수의 라벨된 데이터가 있을때의 모습
- 오른쪽: Decision boundary 근처 데이터를 더 라벨링해서 재학습시, 모델 정확도 향상

Active learning

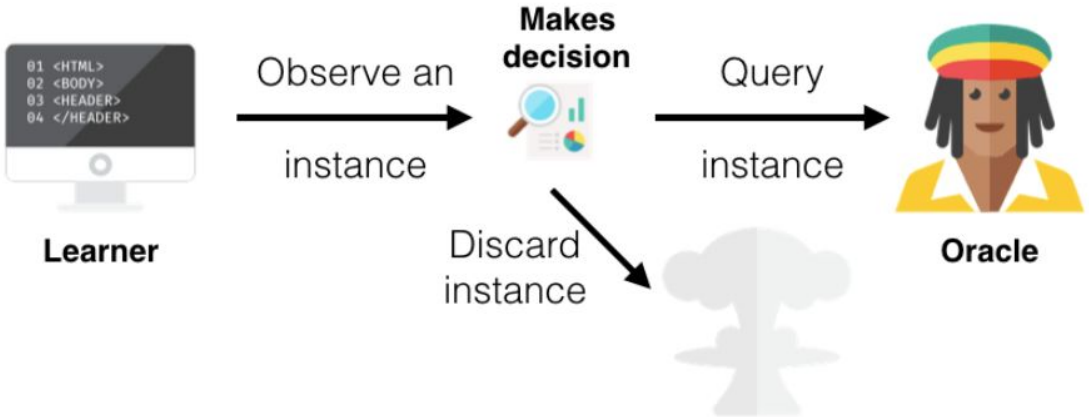
- 레이블링을 점차적으로 할 수 있어 supervised learning보다 효율적
- 전체 데이터중 **중요한 데이터**를 찾아내 가는 것에 포커스
- 이 중요한 데이터를 **어떻게 찾느냐**가 매우 중요

AL: Membership Query Synthesis

현재 데이터의 분포를 기반으로 모델이 레이블해야 할 **데이터를 생성**하고, 이 인스턴스의 라벨링 작업을 전문가에게 요청.

- 손글씨 숫자 데이터 분류 (MNIST) 경우, 손글씨 숫자 이미지를 생성해야 함
- 데이터 생성이 상대적으로 쉬운 경우에 한하여 사용
- NLP에서 Membership Query Synsthesis하는 예: Paper

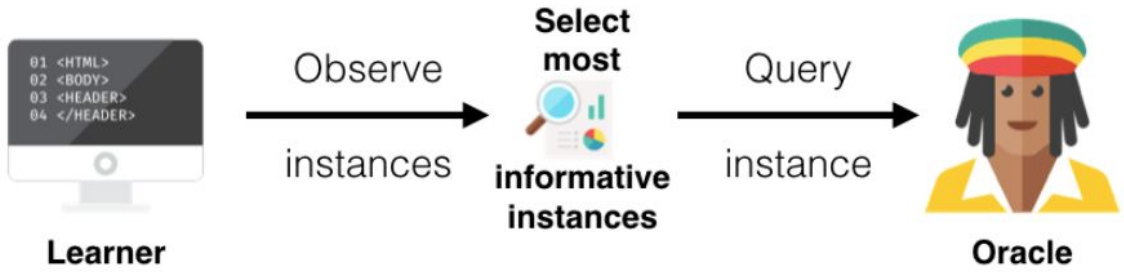
AL: Stream-based Sampling



Label이 없는 데이터 중 하나를 뽑아 “설정 기준"에 따라 전문가에게 라벨링 요청을 할지 결정.

- Unlabeled 데이터조차도 순차적으로 구해야 하는 경우에 사용.

AL: Pool-based sampling



데이터 전체를 보고 어떤 데이터가 중요하고 레이블링 필요한지 판단 (대부분 이 방법 사용)

- Stream-based와 비슷하게 인스턴스 별로 학습에 효율성을 나타내는 지표를 측정하여 가장 효율적인 데이터들을 선별.

AL: Query Strategy

레이블링이 필요한 데이터 선택은 어떻게 할까?

“Cat vs Tiger vs Dog” Classification 모델을 예를 들어보자.

Unlabeled Data	Cat	Tiger	Dog
x1	0.9	0.09	0.01
x2	0.2	0.5	0.3

Reference: *Uncertainty Sampling Cheatsheet*

AL: Query Strategy

Least Confidence (LC):

- 현재 모델에서 가장 확실하게 예측할 수 있는 레이블의 확신도가 가장 낮은 데이터

Unlabeled Data	Cat	Tiger	Dog
x1	0.9	0.09	0.01
x2	0.5	0.2	0.3

x2를 채택

AL: Query Strategy

Margin Sampling:

- 클래스 확률 중, **제일 큰 값과 두번째 값의 차이**가 가장 작은 데이터를 채택.

Unlabeled Data	Cat	Tiger	Dog
x1	0.9	0.09	0.01
x2	0.2	0.5	0.3

x2를 채택

(x1) 0.9 - 0.09 = 0.81
(x2) 0.5 - 0.3 = 0.2

AL: Query Strategy

Ratio-based Sampling:

- 클래스 확률 중, **제일 큰 값과 두번째 값의 비율**이 가장 작은 데이터를 채택.

Unlabeled Data	Cat	Tiger	Dog
x1	0.9	0.09	0.01
x2	0.2	0.5	0.3

x2를 채택

(x1) 0.9 / 0.09 = 10
(x2) 0.5 / 0.3 = 1.66

AL: Query Strategy

Entropy Sampling:

- 모델이 예측한 클래스 확률정보를 모두 사용하여 측정하는 엔트로피가 가장 높은 데이터를 선택.

$$H(x) = -\sum_i p(x_i) \cdot \log p(x_i)$$

(x1): 0.1552
(x2): 0.2698

x2를 채택

Unlabeled Data	Cat	Tiger	Dog
x1	0.9	0.09	0.01
x2	0.2	0.5	0.3

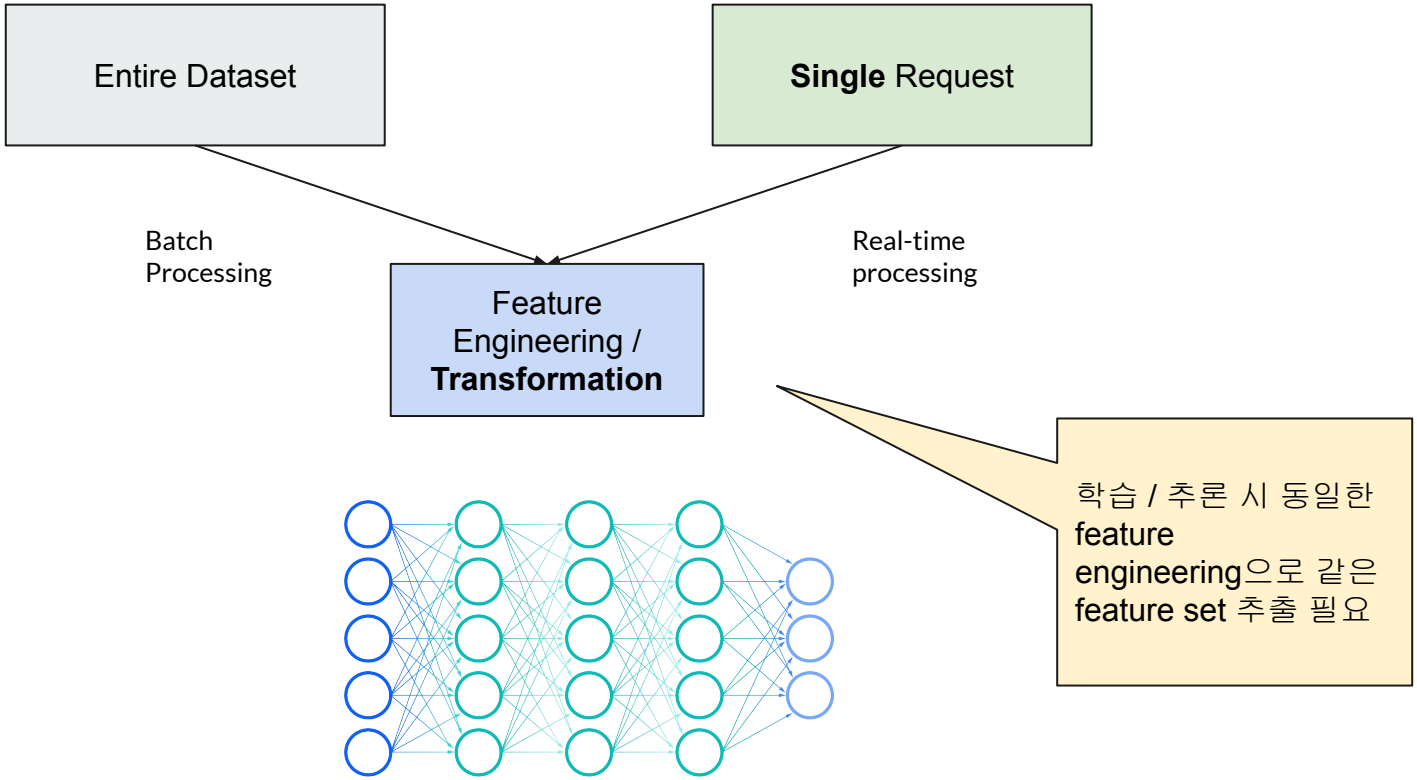
```python
# Evaluating the number of zeros and ones incorrrectly classified
_, _, false_negatives, false_positives = model.evaluate(test_dataset, verbose=0)

print("-" * 100)
print(
    f"Number of zeros incorrectly classified: {false_negatives}, Number of ones incorrectly classified: {false_positives}"
)

# This technique of Active Learning demonstrates ratio based sampling where
# Number of ones/zeros to sample = Number of ones/zeros incorrectly classified / Total incorrectly classified
if false_negatives != 0 and false_positives != 0:
    total = false_negatives + false_positives
    sample_ratio_ones, sample_ratio_zeros = (
        false_positives / total,
        false_negatives / total,
    )
# In the case where all samples are correctly predicted, we can sample both classes equally
else:
    sample_ratio_ones, sample_ratio_zeros = 0.5, 0.5
```

https://github.com/keras-team/keras-io/blob/master/examples/nlp/active_learning_review_classification.py

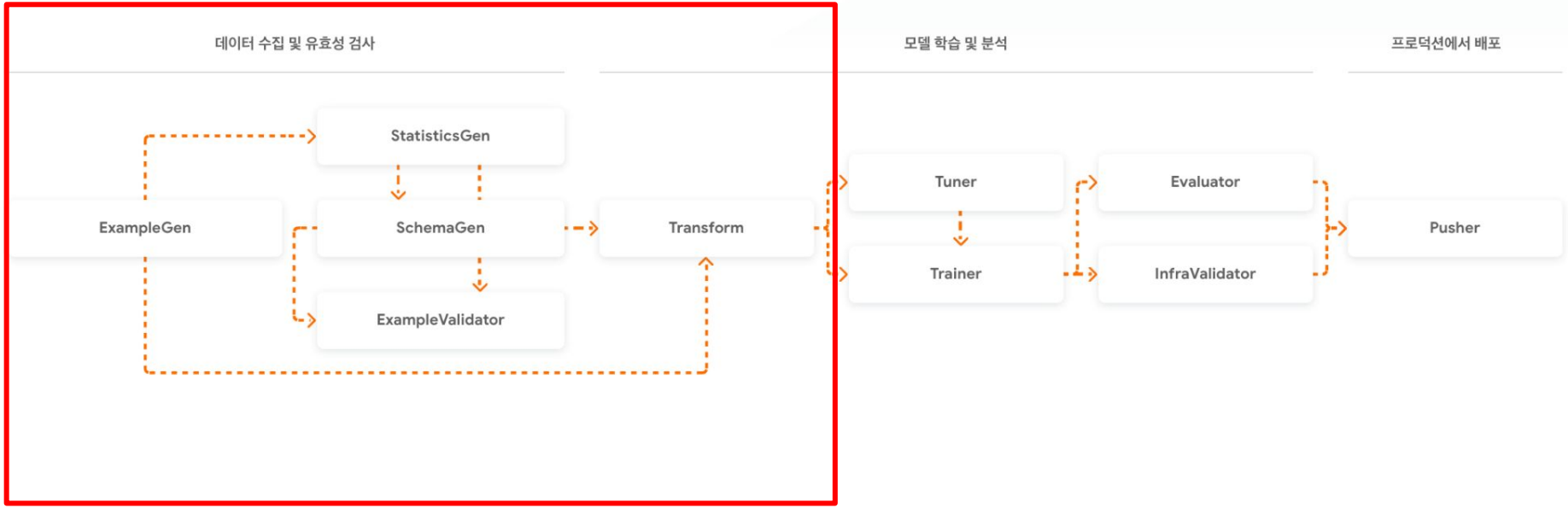
Break

Quiz 5

Feature Transformation in the ML Pipeline

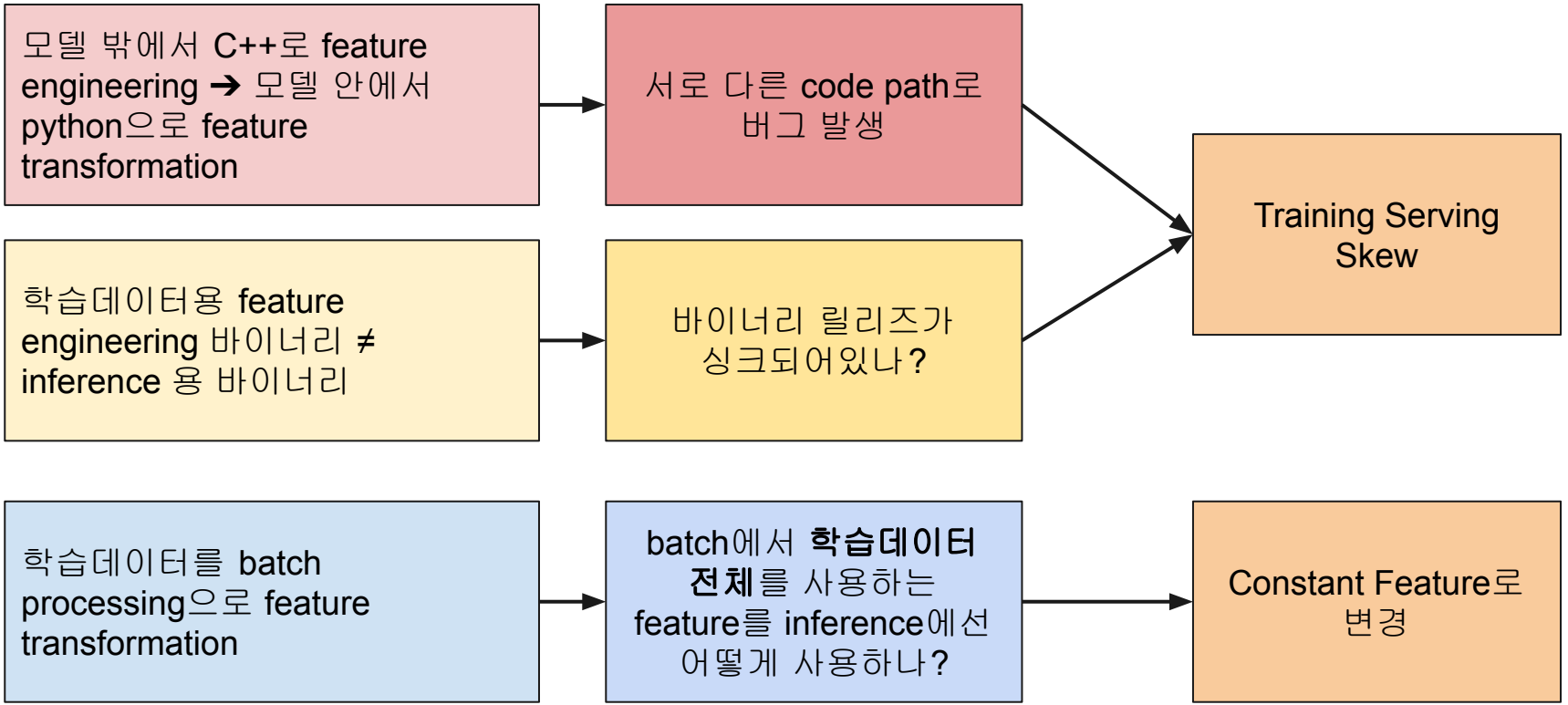

Entire Dataset
Single Request
Batch Processing
Real-time processing
Feature Engineering / Transformation
학습 / 추론 시 동일한 feature engineering으로 같은 feature set 추출 필요

TensorFlow Extended (TFX) Pipeline



전체적인 ML 파이프라인: Data Collection, Feature Engineering, Training, Evaluation, Deployment

Common Pitfalls of Feature Engineering



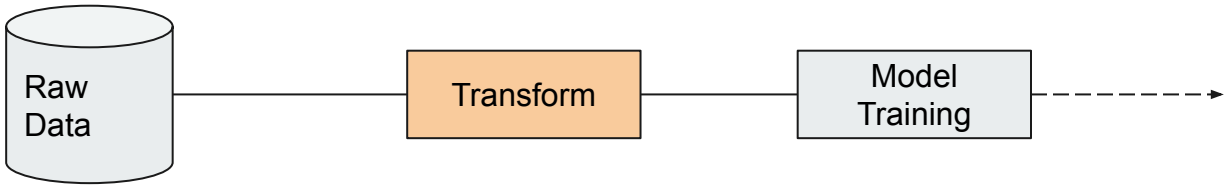
Local vs Global Feature

Local	• Clipping • Multiplying by constant • Feature Expansion	한개의 인풋 인스턴스만으로 계산 가능
Global	• Min • Max • Scaling: (x-mean)/ std • Bucketing	학습데이터 전체를 보고 계산 필요.

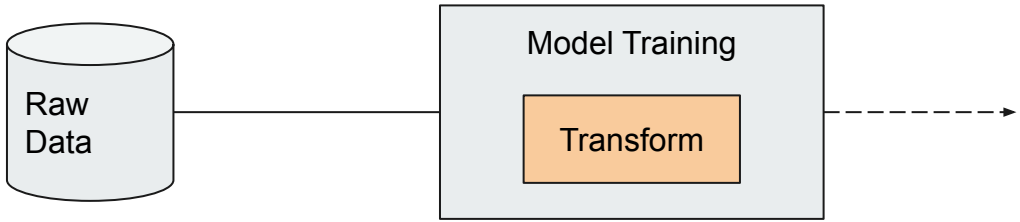
Local vs Global Feature

Local	• Clipping • Multiplying by constant • Feature Expansion	추론 시 (Serving / inference time) 인풋만으로 계산 가능!
Global	• Min • Max • Scaling: (x-mean)/ std • Bucketing	필요한 상수(constant)들을 미리 batch process에서 계산 후 저장 ➔ serving time에 사용.

언제 Feature Transformation을 할것인가?



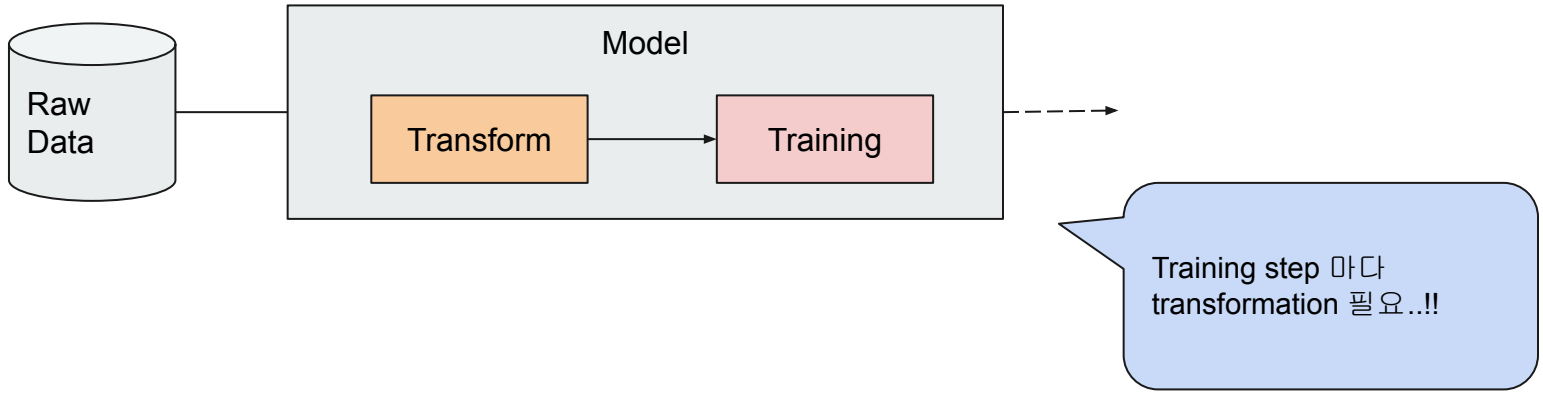
1. Feature Transformation을 모델 "안에서"



모델 "안에서" 계산

장점	• feature 변경해서 iteration하기 용이 • 모델 속에 있기에 train/serving skew가 없음.
단점	• feature transform의 cost가 높음 ➔ inference latency의 증가 • 데이터셋이 전부 필요한 feature를 위한 처리 필요.

모델안에서 feature transform할 경우 최적화 문제

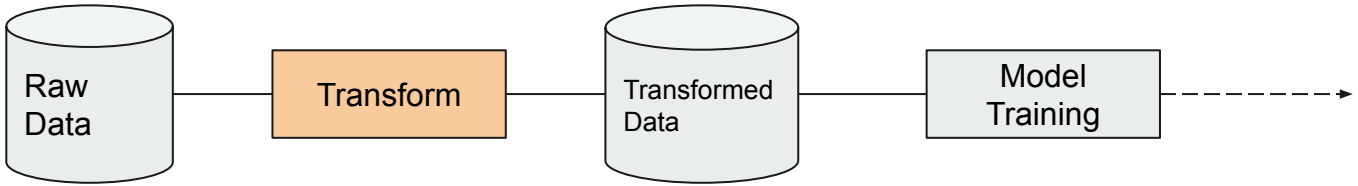


인풋 인스턴스마다
실행되는 Transformation
최적화 필요

Transformation: CPU가 실행
Training: TPU/GPU가 실행

Prefetch:
n 번째 training step 하는동안
n+1 번째 transformation

2. 학습데이터 “전처리" 중 Feature Transformation

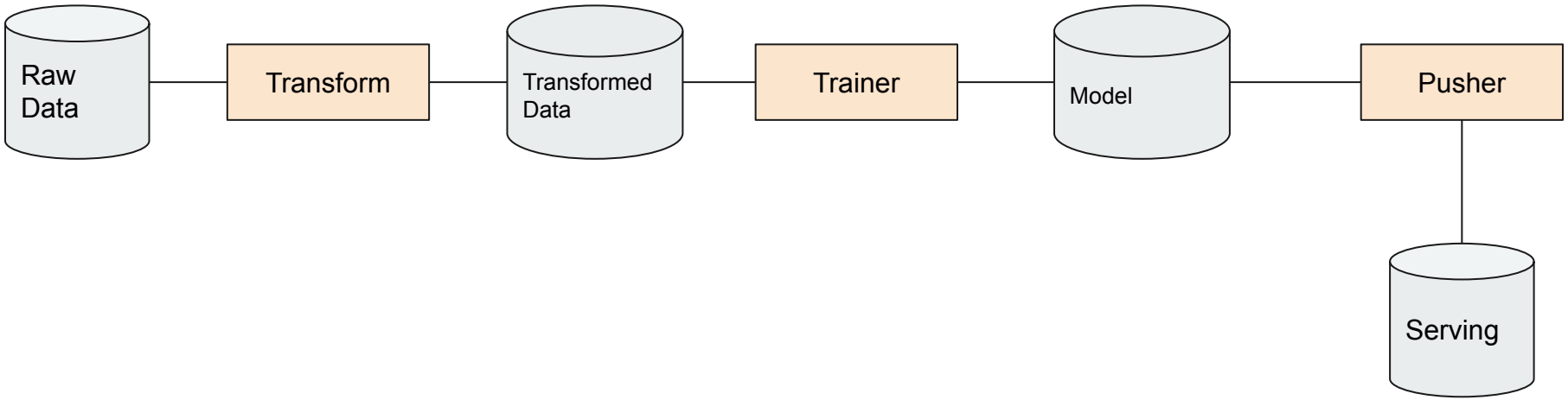


학습데이터의 전처리

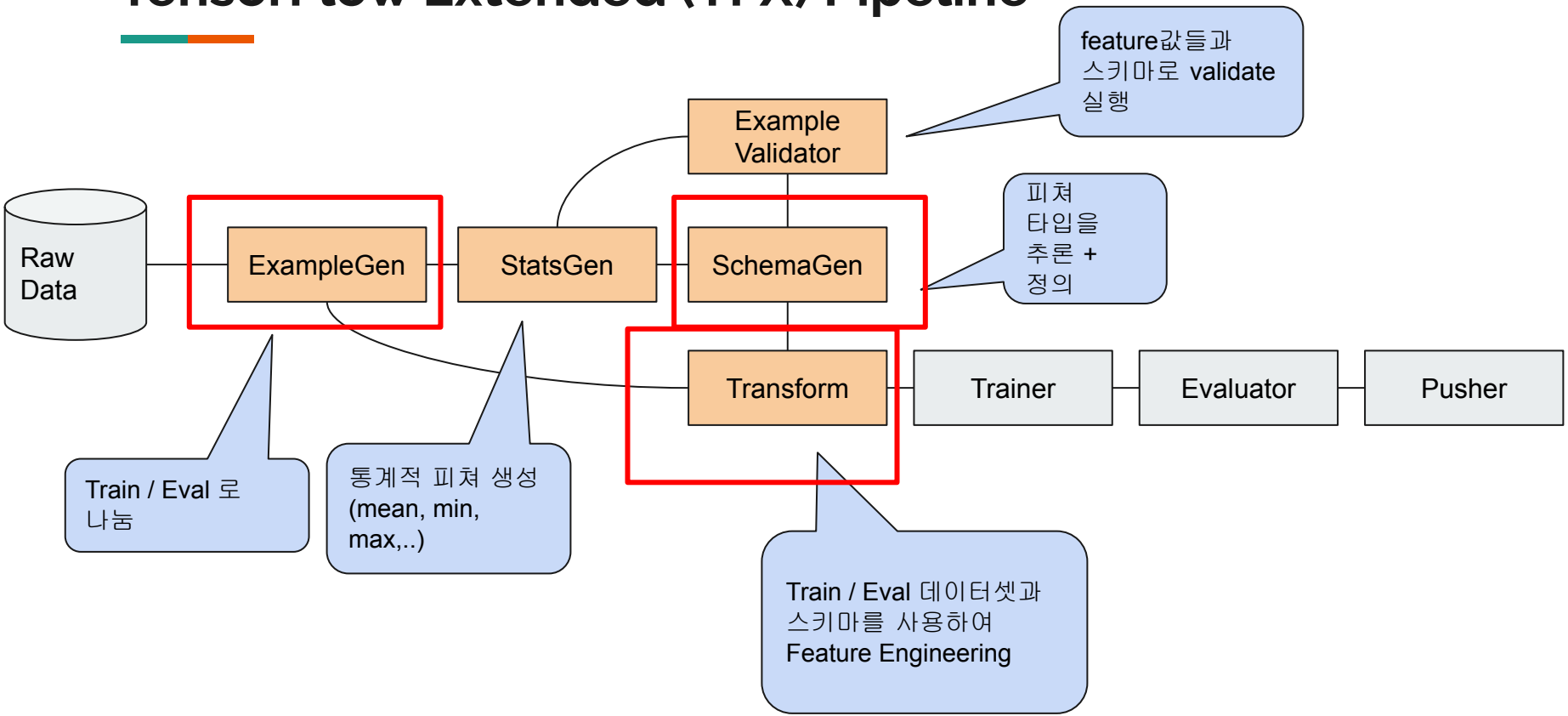
장점	• 학습데이터를 한번 스캔해서 계산 후 저장 가능 • “모델” 변경 시, 학습데이터가 stable.
단점	• feature 변경 할때마다 학습데이터를 • 동일한 전처리를 serving time에도 반복

Feature Transformation Examples: tf.Transformation in TensorFlow

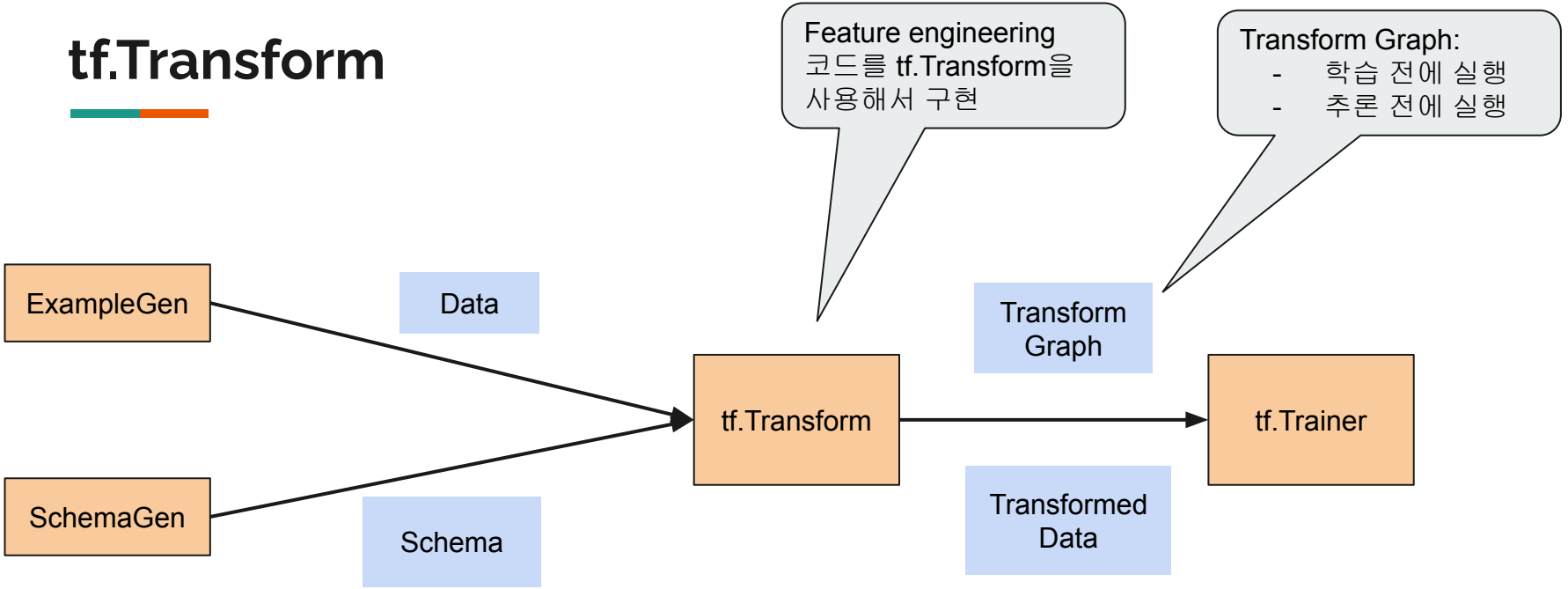
TensorFlow Transform



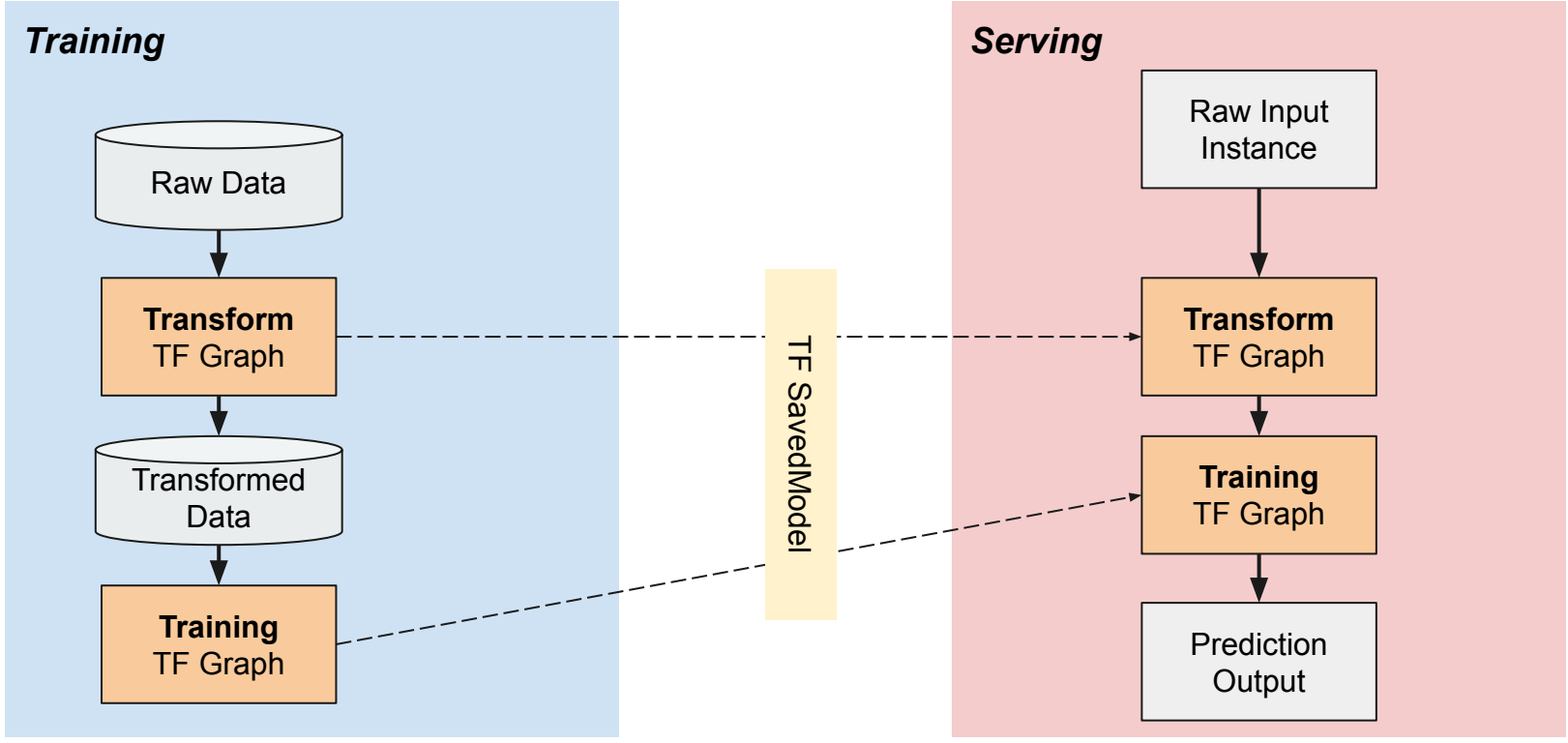
TensorFlow Extended (TFX) Pipeline



tf.Transform



tf.Transform in Training vs Serving



tf.Transform의 기능

- Feature engineering에서 필요한 “operation”들을 **Graph 구조에 저장**.
  - Mean, Max, Stdv
  - String mapping
  - Bucketizing
- 아웃풋 tf.Graph (Transform Graph + Training Graph)에 필요한 모든 constant (상수) feature와 transformation들이 내장
- 학습 (training) 시와 추론 (inference)시 동일한 Transform을 함으로써 Train/Serving skew에서 자유로움.
- 모델 (SavedModel) 자체에 feature transformation을 포함함으로써, 다양한 플랫폼에 deploy해도 동일한 feature engineering을 진행.

코드랩: Preprocess data with TensorFlow Transform

TF Transform 코드랩(기초): (Colab으로 직접 실행가능!)

TF Transform 코드랩 (고급)

TFX History